C 8

```
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
```

```
**FILE**ID**XIDRIVER
```

```
XX      XX    IIIIII    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
XX      XX    IIIIII    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
XX      XX      II      DD      DD  RR      RR      II      VV      VV  EE          RR      RR
XX      XX      II      DD      DD  RR      RR      II      VV      VV  EE          RR      RR
  XX  XX        II      DD      DD  RR      RR      II      VV      VV  EE          RR      RR
  XX  XX        II      DD      DD  RR      RR      II      VV      VV  EE          RR      RR
    XX          II      DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
    XX          II      DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
  XX  XX        II      DD      DD  RR  RR          II      VV      VV  EE          RR  RR
  XX  XX        II      DD      DD  RR  RR          II      VV      VV  EE          RR  RR
XX      XX      II      DD      DD  RR    RR        II        VV  VV    EE          RR    RR
XX      XX      II      DD      DD  RR    RR        II        VV  VV    EE          RR    RR       ....
XX      XX    IIIIII    DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR     ....
XX      XX    IIIIII    DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR     ....
                                                                                                   ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000         .TITLE  XIDRIVER - VAX/VMS DMF32 PARALLEL PORT DRIVER
0000         .IDENT  'V04-001'
0000   ;
0000   ;****************************************************************************
0000   ;*                                                                          *
0000   ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000   ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000   ;*  ALL RIGHTS RESERVED.                                                    *
0000   ;*                                                                          *
0000   ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000   ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000   ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000   ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000   ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000   ;*  TRANSFERRED.                                                            *
0000   ;*                                                                          *
0000   ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000   ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000   ;*  CORPORATION.                                                            *
0000   ;*                                                                          *
0000   ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000   ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000   ;*                                                                          *
0000   ;*                                                                          *
0000   ;****************************************************************************
0000   ;
0000   ;++
0000   ;
0000   ; FACILITY:
0000   ;
0000   ;       VAX/VMS Executive, I/O Drivers
0000   ;
0000   ; ABSTRACT:
0000   ;
0000   ;       This driver is an example driver for the DMF32 parallel port.
0000   ;       This driver implements the DR11C compatibility mode on the device.
0000   ;       It does not implement the silo or DMA options, but serves as a
0000   ;       template to which such features could be added.
0000   ;
0000   ;       This module contains the DMF32 PARALLEL PORT driver:
0000   ;
0000   ;               Tables for loading and dispatching
0000   ;               Controller initialization routine
0000   ;               FDT routine
0000   ;               The start I/O routine
0000   ;               The interrupt service routine
0000   ;               Device specific Cancel I/O
0000   ;
0000   ; ENVIRONMENT:
0000   ;
0000   ;       Kernal Mode, Non-paged
0000   ;
0000   ; AUTHOR:
0000   ;
0000   ;       Jake VanNoy     January 1982
0000   ;
0000   ; MODIFIED BY:
```

```
0000   ;
0000   ;       V04-001 JLV0396         Jake VanNoy             6-SEP-1984
0000   ;               Add AVL to DEVCHAR.
0000   ;
0000   ;       V03-005 JLV0385         Jake VanNoy             23-JUL-1984
0000   ;               Add DPT$M_SVP to DPT.
0000   ;
0000   ;       V03-004 JLV0341         Jake VanNoy             28-MAR-1984
0000   ;               Correct Device IPL.
0000   ;
0000   ;       V03-003 WHM0002         Bill Matthews           16-Feb-1984
0000   ;               Second part of change for edit WHM0001.
0000   ;
0000   ;       V03-002 WHM0001         Bill Matthews           19-Dec-1983
0000   ;               Added code to support new IDB fields IDB$B_COMBO_VECTOR
0000   ;               and IDB$B_COMBO_CSR_OFFSET for determining the main CSR
0000   ;               address and loading the soft vector for the combo device.
0000   ;
0000   ;       V03-001 KDM0002         Kathleen D. Morse       28-Jun-1982
0000   ;               Added $DCDEF and $DYNDEF.
0000   ;
0000   ;--
```

```
0000            .SBTTL  Description of Interface
0000     ;++
0000     ;
0000     ; The DMF32 Parallel Port interface is a 16 bit parallel port for interfacing
0000     ; to a user device. It includes a DR11C compatibility mode (used for word
0000     ; mode within this driver), a silo (buffered) mode (not implemented by this
0000     ; driver), and a DMA mode (also not implemented by this driver). The interface
0000     ; looks like the following:
0000     ;
0000     ;
0000     ;       +--------+                           +--------+
0000     ;       |        |                           |        |
0000     ;       |   D    |---> CTRL 0 ----------->|   U    |
0000     ;       |   M    |---> CTRL 1 ----------->|   S    |
0000     ;       |   F    |                           |   E    |
0000     ;       |   3    |<--- REQ A <------------|   R    |
0000     ;       |   2    |<--- REQ B <------------|        |
0000     ;       |        |                           |   D    |
0000     ;       |   P    |/--- DATA  ------------\|   E    |
0000     ;       |   O    |\--- 16 LINES ---------/|   V    |
0000     ;       |   R    |                           |   I    |
0000     ;       |   T    |---> New Data Ready -->|   C    |  (pulsed on write to OUTBUF)
0000     ;       |        |---> Data Tx'ed ------>|   E    |  (pulsed on read from INBUF)
0000     ;       |        |                           |        |
0000     ;       +--------+                           +--------+
0000     ;
0000     ;
0000     ;--
```

```
0000  ;++
0000  ;
0000  ; This driver may be tested using the following configuration of two DMF32's:
0000  ; The control lines (CTRL 0 and 1) should be tied into request lines (REQ A
0000  ; and B) on the other device. Setting CTRL 0 on the first device causes an
0000  ; interrupt on REQ A on the second device (provided interrupt enable A is set).
0000  ;
0000  ;         +--------+                                             +--------+
0000  ;         :        :                                             :        :
0000  ;         :   D    :-----> CTRL 0 ----------> REQ A ------>:   D    :
0000  ;         :   M    :-----> CTRL 1 ----------> REQ B ------>:   M    :
0000  ;         :   F    :                                             :   F    :
0000  ;         :   3    :<----- REQ A <---------- CTRL 0 <------:   3    :
0000  ;         :   2    :<----- REQ B <---------- CTRL 1 <------:   2    :
0000  ;         :        :                                             :        :
0000  ;         :   P    :/--- DATA  ---------------------------\:   P    :
0000  ;         :   O    :\--- 16 LINES (in each direction) ----/:   O    :
0000  ;         :   R    :                                             :   R    :
0000  ;         :   T    :---> New Data Ready  (not used)              :   T    :
0000  ;         :        :---> Data Tx'ed      (not used)              :        :
0000  ;         :        :                                             :        :
0000  ;         +--------+                                             +--------+
0000  ;
0000  ;--
```

```
0000   .SBTTL  Documentation on interface
0000   ;++
0000   ;
0000   ; The DMF32 parallel port exchanges one 16-bit word at a time.  A single
0000   ; QIO request transfers a buffer of data, with an interrupt requested for
0000   ; each word.
0000   ;
0000   ; For each buffer of data transferred, the DMF32 parallel port allows for
0000   ; the exchange of additional bits of information: the Control and Status
0000   ; Register (CSR) function (CTRL) and status (REQUEST) bits. These bits are
0000   ; accessible to an application process through the device driver QIO
0000   ; interface.  The CTRL bits are labeled CTRL 0 and CTRL 1. The REQUEST
0000   ; bits are labeled REQUEST A and REQUEST B.
0000   ;
0000   ; The user device interfaced to the DMF32 parallel port interprets the
0000   ; value of the two CTRL bits.  The QIO request that initiates the transfer
0000   ; specifies the IO$M_SETFNCT modifer to indicate a change in the value
0000   ; for the CTRL bits. The P4 argument of the request specifies this value.
0000   ; P4 bits 0 and 1 correspond to CTRL bits 0 and 1 respectively. Bits 2
0000   ; through 31 are not used. If required, the CTRL bits must be set for each
0000   ; request. The CTRL bits set in the CSR are passed directly to the user
0000   ; device.
0000   ;
0000   ; The device class for the DMF32 parallel port is DC$_REALTIME and the
0000   ; device type is DT$_XI_DR11C.  The DMF32 parallel port driver does not
0000   ; use the default buffer size field.  The value of this field is set to
0000   ; 65,535. This driver defines no device-dependent characteristics.
0000   ;
0000   ; The DMF32 parallel port can perform logical, virtual, and physical I/O
0000   ; operations. The basic I/O functions are read, write, set mode, and set
0000   ; characteristics.
0000   ;
0000   ; +-----------------------+-----------------+------------------+
0000   ; !  Function Code and    !    Function     !     Function     !
0000   ; !     Arguments         !    Modifiers    !                  !
0000   ; !-----------------------+-----------------+------------------!
0000   ; !                       !                 !                  !
0000   ; ! IO$_READLBLK P1,P2,-  ! IO$M_SETFNCT    ! Read block       !
0000   ; !              P3,P4    ! IO$M_RESET      !                  !
0000   ; !                       ! IO$M_TIMED      !                  !
0000   ; !                       !                 !                  !
0000   ; ! IO$_WRITELBLK P1,P2,- ! IO$M_SETFNCT    !Write logical block!
0000   ; !               P3,P4   ! IO$M_RESET      !                  !
0000   ; !                       ! IO$M_TIMED      !                  !
0000   ; !                       !                 !                  !
0000   ; ! IO$_SETMODE P1,P3     ! IO$M_ATTNAST    !Set PORT charact- !
0000   ; !                       !                 !eristics for subse-!
0000   ; !                       !                 ! quent operations !
0000   ; !                       !                 !                  !
0000   ; ! IO$_SETCHAR P1,P3     ! IO$M_ATTNAST    !Set PORT charact- !
0000   ; !                       !                 !eristics for subse-!
0000   ; !                       !                 ! quent operations !
0000   ; !                       !                 !                  !
0000   ; +-----------------------+-------+---------------+-----------------+
0000   ;
0000   ; Not in above table are functions IO$_READPBLK, IO$_READVBLK, WRITEPBLK
0000   ; and WRITELBLK. There is no functional difference in these operations.
```

```
0000  ; Although the DMF32 parallel port does not differentiate between logical,
0000  ; virtual, and physical I/O functions (all are treated identically), the
0000  ; user must have the required privilege to issue a request.
0000  ;
0000  ; The function-dependent arguments for the read and write function codes are:
0000  ;
0000  ; o     P1 -- the starting virtual address of the buffer that
0000  ;       is to receive data in the case of a read operation; or, in
0000  ;       the case of a write operation, the virtual address of the
0000  ;       buffer that is to send data to the DMF32 parallel port.
0000  ;       Modify access to the buffer, rather than read or write
0000  ;       access, is checked for all block mode read and write
0000  ;       requests.
0000  ;
0000  ; o     P2 -- the size of the data buffer in bytes, that is, the
0000  ;       transfer count.  Since the DMF32 parallel port performs
0000  ;       word transfers, the transfer count must be an even value.
0000  ;
0000  ; o     P3 -- the timeout period for this request (in seconds).
0000  ;       The value specified must be equal to or greater than 2.
0000  ;       IO$M_TIMED must be specified.  The default timeout value for each
0000  ;       request is 10 seconds.
0000  ;
0000  ; o     P4 -- the value of the DMF32 parallel port Command and Status
0000  ;       Register (CSR) function (CTRL) bits to be set.  If
0000  ;       IO$M_SETFNCT is specified, the low-order three bits of P4
0000  ;       (2:0) are written to CSR CTRL bits 1:0 (respectively) at the
0000  ;       time of transfer.
0000  ;
0000  ; The transfer count specified by the P2 argument must be an even number
0000  ; of bytes.  If an odd number or more than 65534 bytes is specifed, an
0000  ; error (SS$_BADPARAM) is returned in the I/O status block (IOSB). If the
0000  ; transfer count is 0, the driver will transfer no data.  However, if
0000  ; IO$M_SETFNCT is specified and P2 is 0, the driver will set the CTRL bits
0000  ; in the DMF32 parallel port CSR, and return the current CSR status bit
0000  ; values in the IOSB.
0000  ;
0000  ; The read and write QIO functions can take three function modifiers:
0000  ;
0000  ; o     IO$M_SETFNCT - set the function (CTRL) bits in the DMF32 parallel
0000  ;       port CSR before the data transfer is initiated.  The
0000  ;       low-order two bits of the P4 argument specify the CTRL
0000  ;       bits.  The user device that interfaces the DMF32 PARALLEL
0000  ;       PORT receives the CTRL bits directly and their value is
0000  ;       interpreted entirely by the device.
0000  ;
0000  ; If an unsolicited interrupt is received from the DMF32 parallel port, no
0000  ; read or write request is posted, and the next request is for a word mode
0000  ; read, the driver will return the word read from the DMF32 parallel port
0000  ; INBUF and store it in the first word of the user's buffer. In this case
0000  ; the driver does not wait for an interrupt.
0000  ;
0000  ; o     IO$M_TIMED - set the device timeout interval for the data
0000  ;       transfer request. The P3 argument specifies the timeout
0000  ;       interval value in seconds.  For consistent results, this
0000  ;       value must be equal to or greater than 2.
0000  ;
```

```
0000    ; o     IO$M_RESET - perform a device reset to the DMF32 parallel port before
0000    ;       any I/O operation is initiated. This function does not
0000    ;       affect any other device on the system or on the DMF32.
0000    ;
0000    ; The set mode and characteristic function codes are:
0000    ;
0000    ; o     IO$_SETMODE
0000    ;
0000    ; o     IO$_SETCHAR
0000    ;
0000    ; These functions take the following device/function-dependent arguments:
0000    ;
0000    ; o     P1 - the virtual address of a quadword characteristics buffer.  If
0000    ;       the function modifer IO$M_ATTNAST is specified, P1 is the
0000    ;       address the AST service routine. In this case, if P1 is 0,
0000    ;       all attention ASTs are disabled.
0000    ;
0000    ; o     P3 - the access mode to deliver the AST (maximized with the
0000    ;       requestor's access mode).  If IO$M_ATTNAST is not specified, P3
0000    ;       is ignored.
0000    ;
0000    ;       Figure 3-4 shows the quadword P1 characteristics buffer for
0000    ;       IO$_SETMODE and IO$_SETCHAR.
0000    ;
0000    ;       31                      16 15          8 7           0
0000    ;       +-------------------------+-------------+-------------+
0000    ;       !                         !             !             !
0000    ;       !        not used         !    type     !    class    !
0000    ;       !                         !             !             !
0000    ;       +-------------------------+-------------+-------------+
0000    ;       !                                                     !
0000    ;       !                device characteristics               !
0000    ;       !                                                     !
0000    ;       +-----------------------------------------------------+
0000    ;
0000    ; The IO$_SETMODE and IO$_SETCHAR function codes can take the following function
0000    ; modifier:
0000    ;
0000    ; o     IO$M_ATTNAST - enable attention AST
0000    ;
0000    ; This function modifier allows the user process to queue an attention AST
0000    ; for delivery when an asynchronous or unsolicited condition is detected
0000    ; by the DMF32 parallel port driver.  Unlike ASTs for other QIO functions,
0000    ; use of this function modifier does not increment the I/O count for the
0000    ; requesting process or lock pages in memory for I/O buffers. There must
0000    ; be an AST quota for each AST.
0000    ;
0000    ; Attention ASTs are delivered under the following conditions:
0000    ;
0000    ; o     An unsolicited interrupt from the DMF32 parallel port occurs.
0000    ;
0000    ; o     An attention AST is queued and a previous unsolicited interrupt
0000    ;       has not been acknowledged.
0000    ;
0000    ; The $CANCEL system service is used to flush attention ASTs for a specific
0000    ; channel.
0000    ;
```

```
0000  ; IO$_SETMODE!IO$M_ATTNAST and IO$_SETCHAR!IO$M_ATTNAST are one-time AST
0000  ; enables; they must be explicitly re-enabled once the AST has been
0000  ; delivered if the user desires notification of the next interrupt.  Use
0000  ; of this function modifier does not update the device characteristics.
0000  ;
0000  ; After the AST is delivered, the QIO astprm parameter contains the
0000  ; contents of the DMF32 parallel port CSR in the low two bytes and the
0000  ; value read from the DMF32 parallel port INBUF in the high two bytes.
0000  ;
0000  ; On completion of each read or write request, the I/O status block
0000  ; is filled with system and DMF32 parallel port status information.
0000  ;
0000  ;                                +2                        IOSB
0000  ;         +-------------------------+---------------------------+
0000  ;         :                         :                           :
0000  ;         :       byte count        :          status           :
0000  ;         :                         :                           :
0000  ;         +-------------------------+---------------------------+
0000  ;         :                         :                           :
0000  ;         :       unused            :         PORT CSR          :
0000  ;         :                         :                           :
0000  ;         +-------------------------+---------------------------+
0000  ;
0000  ;
0000  ;--
0000
```

```
                  0000          .SBTTL  External and local symbol definitions
                  0000
                  0000
                  0000  ; External symbols
                  0000
                  0000          $ACBDEF                                 ; AST control block
                  0000          $CRBDEF                                 ; Channel request block
                  0000          $DCDEF                                  ; Device types
                  0000          $DDBDEF                                 ; Device data block
                  0000          $DPTDEF                                 ; Driver prolog table
                  0000          $DYNDEF                                 ; Dynamic data structure types
                  0000          $IDBDEF                                 ; Interrupt data block
                  0000          $IODEF                                  ; I/O function codes
                  0000          $IPLDEF                                 ; Hardware IPL definitions
                  0000          $IRPDEF                                 ; I/O request packet
                  0000          $PRDEF                                  ; Internal processor registers
                  0000          $PRIDEF                                 ; Scheduler priority increments
                  0000          $SSDEF                                  ; System messages
                  0000          $UCBDEF                                 ; Unit control block
                  0000          $VECDEF                                 ; Interrupt vector block
                  0000
                  0000  ; Local symbols
                  0000
                  0000  ; Argument list (AP) offsets for device-dependent QIO parameters
                  0000
        00000000  0000  P1      = 0                                     ; First QIO parameter
        00000004  0000  P2      = 4                                     ; Second QIO parameter
        00000008  0000  P3      = 8                                     ; Third QIO parameter
        0000000C  0000  P4      = 12                                    ; Fourth QIO parameter
        00000010  0000  P5      = 16                                    ; Fifth QIO parameter
        00000014  0000  P6      = 20                                    ; Sixth QIO parameter
                  0000
                  0000  ; Other constants
                  0000
        0000000A  0000  XI_DEF_TIMEOUT  = 10                            ; 10 second default device timeout
        0000FFFF  0000  XI_DEF_BUFSIZ   = 65535                         ; Default buffer size
        00000002  0000  XI$K_VEC_OFFSET = 2                             ; Vector offset
                  0000
                  0000  ;
                  0000  ; Macros
                  0000  ;
                  0000
                  0000  ;
                  0000  ; The SETCTRL macro sets the CTRL 0 and 1 lines as they have been
                  0000  ; specified in P4 in a read or write QIO. They are cleared and a wait
                  0000  ; occurs before being set. This is because testing for this example
                  0000  ; driver was done between two DMF32's in word mode, and the delay is so the
                  0000  ; microcode on the DMF32 can see the control line changes.
                  0000  ;
                  0000
                  0000  .MACRO  SETCTRL
                  0000          BICW    #XI_CSR$M_CTRL0!XI_CSR$M_CTRL1,XI_CSR(R4)
                  0000          CLRL    -(SP)
                  0000          TIMEWAIT -
                  0000                  TIME = #2,-
                  0000                  BITVAL = #1,-
                  0000                  SOURCE = (SP),-
```

```
                                   0000   390                    CONTEXT = L,-
                                   0000   391                    SENSE = .TRUE.
                                   0000   392             TSTL   (SP)+
                                   0000   393             BISW   IRP$L_SEGVBN(R3),XI_CSR(R4)
                                   0000   394 .ENDM       SETCTRL
```

```
                 0000
                 0000    ; UCB_XI definitions that follow the standard UCB fields
                 0000
                 0000            $DEFINI UCB
                 0000
       000000A0  0000            .=UCB$L_DPC+4   ;
                 00A0
                 00A0    $DEF    UCB$L_XI_ATTN
       000000A4  00A0                            .BLKL   1       ; Attention AST queue
                 00A4
                 00A4    $DEF    UCB$L_XI_DPR
       000000A8  00A4                            .BLKL   1       ; Word count?
                 00A8
                 00A8    $DEF    UCB$W_XI_INBUF
       000000AA  00A8                            .BLKW   1       ; Input buffer temporary
                 00AA
                 00AA    $DEF    UCB$W_XI_CSR
       000000AC  00AA                            .BLKW   1       ; CSR temporary
                 00AC
                 00AC    ; Bit positions for device-dependent status field in UCB (UCB$W_DEVSTS)
                 00AC
                 00AC            $VIELD  UCB,0,<-                ; UCB device specific bit definitions
                 00AC                    <ATTNAST,,M>,-
                 00AC                    <UNEXPT,,M>-
                 00AC                    >
                 00AC
       000000AC  00AC    UCB$K_SIZE=.
                 00AC            $DEFEND UCB
                 0000   ;
                 0000   ; DMF32 Parallel Port CSR definitions
                 0000   ;
                 0000           $DEFINI XI
                 0000
                 0000   $DEF    XI_CSR                          ; Device CSR
                 0000
                 0000   ; Bit positions for device control/status register
                 0000
                 0000           $VIELD  XI_CSR,0,<-             ; Control/status register
                 0000                   <CTRL0,,M>,-            ; Control line 0
                 0000                   <CTRL1,,M>,-            ; Control line 1
                 0000                   <NPR_PS,,M>,-           ; NPR Primary/Secondary
                 0000                   <INDREG,2,M>,-          ; Indirect Register Address
                 0000                   <INTENB_A,,M>,-         ; Interrupt Enable A
                 0000                   <INTENB_B,,M>,-         ; Interrupt Enable B
                 0000                   <REQ_A,,M>,-            ; Request A
                 0000                   <DONE_P,,M>,-           ; Done Primary
                 0000                   <DONE_S,,M>,-           ; Done Secondary
                 0000                   <,,M>,-                 ; unused
                 0000                   <FLUSH,,M>,-            ; Flush Buffer
                 0000                   <,,M>,-                 ; unused
                 0000                   <NXMERR,,M>,-           ; Non-existent memory error
                 0000                   <RESET,,M>,-            ; Master Reset
                 0000                   <REQ_B,,M>-             ; Request B
                 0000           >
                 0000
        00000060 0000   XI_CSR$M_IEAB   = <XI_CSR$M_INTENB_A>!<XI_CSR$M_INTENB_B> ; Interrupt enable mask
                 0000
        00000002 0000                           .BLKW   1
                 0002   $DEF    XI_OUTBUF                       ; Output buffer Register
        00000004 0002                           .BLKW   1
                 0004
                 0004   ; Note that XI_INBUF and XI_MISC are at the same offset
                 0004
                 0004   $DEF    XI_INBUF                        ; Input buffer Register (when read)
                 0004   $DEF    XI_MISC                         ; Miscellaneous Register (when written)
                 0004
                 0004   ; Bit positions for miscellaneous register
                 0004
                 0004           $VIELD  XI_MISC,0,<-            ; Miscellaneous register
                 0004                   <MODE,4,M>,-            ; Mode
                 0004                   <,10,M>,-               ; unused
                 0004                   <SECBUF,,M>-            ; Secondary Buffer Address, Bit 17
                 0004                   <PRIBUF,,M>-            ; Primary Buffer Address, Bit 17
                 0004           >
        00000006 0004                           .BLKW   1
                 0006   $DEF    XI_IND                  .BLKW   1       ; Indirect Register
        00000008 0006                           .BLKW   1
                 0008
                 0008           $DEFEND XI                      ; End of PORT CSR definitions
```

```
0000          .SBTTL  Device Driver Tables
0000
0000   ; Driver prologue table
0000
0000          DPTAB   -                               ; DPT-creation macro
                      END=XI_END,-                    ; End of driver label
0000                  ADAPTER=UBA,-                   ; Adapter type
0000                  FLAGS=DPT$M_SVP,-               ; Allocate system page table
0000                  UCBSIZE=UCB$K_SIZE,-            ; UCB size
0000                  NAME=XIDRIVER                   ; Driver name
0038
0038          DPT_STORE INIT                          ; Start of load
0038                                                  ; initialization table
0038          DPT_STORE UCB,UCB$B_FIPL,B,8            ; Device fork IPL
003C          DPT_STORE UCB,UCB$B_DIPL,B,21           ; Device interrupt IPL
0040          DPT_STORE UCB,UCB$L_DEVCHAR,L,<-        ; Device characteristics
0040                  DEV$M_AVL!-                     ; Available
0040                  DEV$M_RTM!-                     ; Real Time device
0040                  DEV$M_IDV!-                     ;   input device
0040                  DEV$M_ODV>                      ;   output device
0047          DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_REALTIME      ; Device class
004B          DPT_STORE UCB,UCB$B_DEVTYPE,B,DT$_XI_DR11C       ; Device Type
004F          DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,-       ; Default buffer size
004F                  XI_DEF_BUFSIZ
0054          DPT_STORE REINIT                        ; Start of reload
0054                                                  ; initialization table
0054          DPT_STORE DDB,DDB$L_DDT,D,XI$DDT        ; Address of DDT
0059          DPT_STORE CRB,CRB$L_INTD+4,D,-          ; Address of interrupt
0059                  XI_INTERRUPT                    ; service routine
005E          DPT_STORE CRB,CRB$L_INTD2+4,D,-         ; Address of interrupt
005E                  XI_INTERRUPT                    ; service routine
0063          DPT_STORE CRB,CRB$L_INTD+VEC$L_INITIAL,-; Address of controller
0063                  D,XI_CONTROL_INIT               ; initialization routine
0068          DPT_STORE END                           ; End of initialization
0000                                                  ; tables
0000
0000   ; Driver dispatch table
0000
0000          DDTAB   -                               ; DDT-creation macro
0000                  DEVNAM=XI,-                     ; Name of device
0000                  START=XI_START,-                ; Start I/O routine
0000                  FUNCTB=XI_FUNCTABLE,-           ; FDT address
0000                  CANCEL=XI_CANCEL                ; Cancel I/O routine
```

```
0038    ;
0038    ; Function dispatch table
0038    ;
0038    XI_FUNCTABLE:                                           ; FDT for driver
0038
0038            ; Valid I/O functions
0038
0038            FUNCTAB ,-
0038                    <READPBLK,READLBLK,READVBLK, -
0038                    WRITEPBLK,WRITELBLK,WRITEVBLK,-
0038                    SETMODE,SETCHAR,SENSEMODE,SENSECHAR>
0040
0040            FUNCTAB ,                               ; No buffered functions
0048
0048            FUNCTAB XI_READ_WRITE,-                         ; Device-specific FDT
0048                    <READPBLK,READLBLK,READVBLK, -
0048                    WRITEPBLK,WRITELBLK,WRITEVBLK>
0054            FUNCTAB +EXE$READ,<READPBLK,READLBLK,READVBLK>
0060            FUNCTAB +EXE$WRITE,<WRITEPBLK,WRITELBLK,WRITEVBLK>
006C            FUNCTAB XI_SETMODE,<SETMODE,SETCHAR>
0078            FUNCTAB +EXE$SENSEMODE,<SENSEMODE,SENSECHAR>
```

```
                         0084          .SBTTL  XI_CONTROL_INIT, Controller initialization
                         0084
                         0084  ;++
                         0084  ; XI_CONTROL_INIT, Called when driver is loaded, system is booted, or
                         0084  ; power failure recovery.
                         0084  ;
                         0084  ; Functional Description:
                         0084  ;
                         0084  ;       1) Allocates the direct data path permanently
                         0084  ;       2) Assigns the controller data channel permanently
                         0084  ;       3) Clears the Control and Status Register
                         0084  ;       4) If power recovery, requests device time-out
                         0084  ;
                         0084  ; Inputs:
                         0084  ;
                         0084  ;       R4 = address of CSR
                         0084  ;       R5 = address of IDB
                         0084  ;       R6 = address of DDB
                         0084  ;       R8 = address of CRB
                         0084  ;
                         0084  ; Outputs:
                         0084  ;
                         0084  ;       VEC$V_PATHLOCK bit set in CRB$L_INTD+VEC$B_DATAPATH
                         0084  ;       UCB address placed into IDB$L_OWNER
                         0084  ;
                         0084  ;
                         0084  ;--
                         0084
                         0084  XI_CONTROL_INIT:
                         0084
      50    18 A5    D0  0084          MOVL    IDB$L_UCBLST(R5),R0     ; Address of UCB
      04 A5    50    D0  0088          MOVL    R0,IDB$L_OWNER(R5)      ; Make permanent controller owner
      64 A0    10    A8  008C          BISW    #UCB$M_ONLINE, -
                         0090                  UCB$W_STS(R0)           ; Set device status "on-line"
                         0090
                         0090  ; If powerfail has occured and device was active, force device time-out.
                         0090  ; The user can set his own time-out interval for each request.  Time-
                         0090  ; out is forced so a very long time-out period will be short circuited.
                         0090
   05 64 A0    05    E0  0090          BBS     #UCB$V_POWER, -
                         0095                  UCB$W_STS(R0),10$       ; Branch if powerfail
   37 A8    80 8F    88  0095          BISB    #VEC$M_PATHLOCK, -
                         009A                  CRB$L_INTD+VEC$B_DATAPATH(R8) ; Permanently allocate direct datapath
                         009A  10$:
                         009A
      50    0F A5    98  009A          CVTBL   IDB$B_COMBO_CSR_OFFSET(R5),R0    ; GET OFFSET TO MAIN DMF CSR
            10 A5    83  009E          SUBB3   IDB$B_COMBO_VECTOR_OFFSET(R5),- ; CALCULATE AND LOAD THE
    6440    0B A5        00A1                  IDB$B_VECTOR(R5),(R4)[R0]       ;         VECTOR ADDRESS
            030D     30  00A5          BSBW    XI_DEV_RESET                    ; Reset port
                     05  00A8          RSB                                     ; Done
```

```
                                  00A9              .SBTTL  XI_READ_WRITE,   Data transfer FDT
                                  00A9
                                  00A9      ;++
                                  00A9      ; XI_READ_WRITE, FDT for READLBLK,READVBLK,READPBLK,WRITELBLK,WRITEVBLK,
                                  00A9      ;                        WRITEPBLK
                                  00A9      ;
                                  00A9      ; Functional description:
                                  00A9      ;
                                  00A9      ;       1) Rejects QUEUE I/O's with odd transfer count
                                  00A9      ;       2) Rejects QUEUE I/O's for DMA request to UBA Direct Data
                                  00A9      ;          PATH on odd byte boundary
                                  00A9      ;       3) Stores request time-out count specified in P3 into IRP
                                  00A9      ;       4) Stores CTRL bits specified in P4 into IRP
                                  00A9      ;       6) Checks block mode transfers for memory modify access
                                  00A9      ;
                                  00A9      ; Inputs:
                                  00A9      ;
                                  00A9      ;       R3 = Address of IRP
                                  00A9      ;       R4 = Address of PCB
                                  00A9      ;       R5 = Address of UCB
                                  00A9      ;       R6 = Address of CCB
                                  00A9      ;       AP = Address of P1
                                  00A9      ;               P1 = Buffer Address
                                  00A9      ;               P2 = Buffer size in bytes
                                  00A9      ;               P3 = Request time-out period (conditional on IO$M_TIMED)
                                  00A9      ;               P4 = Value for CSR CTRL bits (conditional on IO$M_SETFNCT)
                                  00A9      ;               P5 = 0 for Request A, 1 for Request B (DMA)
                                  00A9      ;
                                  00A9      ; Outputs:
                                  00A9      ;
                                  00A9      ;       R0 = Error status if odd transfer count
                                  00A9      ;       IRP$L_MEDIA = Time-out count for this request
                                  00A9      ;       IRP$L_SEGVBN = CTRL bits for PORT CSR
                                  00A9      ;
                                  00A9      ;--
                                  00A9
                                  00A9      XI_READ_WRITE:
                                  00A9
          09 04 AC         E9     00A9              BLBC    P2(AP),20$              ; Branch if transfer count even
             50   14       3C     00AD      10$:    MOVZWL  #SS$_BADPARAM,R0        ; Set error status code
        00000000'GF        17     00B0              JMP     G^EXE$ABORTIO           ; Abort request
                                  00B6
          51   20 A3       3C     00B6      20$:    MOVZWL  IRP$W_FUNC(R3),R1       ; Fetch I/O Function code
       38 A3   08 AC       D0     00BA              MOVL    P3(AP),IRP$L_MEDIA(R3)  ; Set request specific time-out count
       04 51      07       E0     00BF              BBS     #IO$V_TIMED,R1,30$      ; Branch if time-out specified
          38 A3   0A       3C     00C3              MOVZWL  #XI_DEF_TIMEOUT, -
                                  00C7                      IRP$L_MEDIA(R3)         ; Else set default timeout value
   0C AC    02    00       EF     00C7      30$:    EXTZV   #0,#2,P4(AP),-
             48 A3                00CC                      IRP$L_SEGVBN(R3)        ; Get value for CTRL bits
                           05     00CE              RSB                             ; Return
```

```
                            00CF        .SBTTL  XI_SETMODE,      Set Mode, Set Char FDT
                            00CF
                            00CF  ;++
                            00CF  ; XI_SETMODE, FDT routine to process SET MODE and SET CHARACTERISTICS
                            00CF  ;
                            00CF  ; Functional description:
                            00CF  ;
                            00CF  ;       If IO$M_ATTNAST modifier is set, queue attention AST for device
                            00CF  ;       Else, finish I/O.
                            00CF  ;
                            00CF  ; Inputs:
                            00CF  ;
                            00CF  ;       R3 = I/O packet address
                            00CF  ;       R4 = PCB address
                            00CF  ;       R5 = UCB address
                            00CF  ;       R6 = CCB address
                            00CF  ;       R7 = Function code
                            00CF  ;       AP = QIO Paramater list address
                            00CF  ;
                            00CF  ; Outputs:
                            00CF  ;
                            00CF  ;       If IO$M_ATTNAST is specified, queue AST on UCB attention AST list.
                            00CF  ;       Else, use exec routine to update device characteristics
                            00CF  ;
                            00CF  ;--
                            00CF
                            00CF  XI_SETMODE:
                            00CF
   50   20 A3     3C        00CF        MOVZWL  IRP$W_FUNC(R3),R0       ; Get entire function code
   28 50    08    E1        00D3        BBC     #IO$V_ATTNAST,R0,20$    ; Branch if not an ATTN AST
                            00D7
                            00D7  ; Attention AST request
                            00D7
       0090 8F    BB        00D7        PUSHR   #^M<R4,R7>
  57    00A0 C5   9E        00DB        MOVAB   UCB$L_XI_ATTN(R5),R7    ; Address of ATTN AST control block list
   00000000'GF    16        00E0        JSB     G^COM$SETATTNAST        ; Set up attention AST
       0090 8F    BA        00E6        POPR    #^M<R4,R7>
         18 50    E9        00EA        BLBC    R0,30$                  ; Branch if error
   68 A5    01    A8        00ED        BISW    #UCB$M_ATTNAST, -
                            00F1                UCB$W_DEVSTS(R5)        ; Flag ATTN AST expected.
 03 68 A5   01    E1        00F1        BBC     #UCB$V_UNEXPT, -
                            00F6                UCB$W_DEVSTS(R5),10$    ; Deliver AST if unsolicited interrupt
          026D    30        00F6        BSBW    XI_DEL_ATTNAST
   00000000'GF    17        00F9  10$:  JMP     G^EXE$FINISHIO          ; Thats all for now
                            00FF
   00000000'GF    17        00FF  20$:  JMP     G^EXE$SETCHAR           ; Set device characteristics
                            0105
            51    D4        0105  30$:  CLRL    R1                      ; zero R1
   00000000'GF    17        0107        JMP     G^EXE$ABORTIO           ; Abort I/O with R0 as status
                            010D
```

```
                        010D            .SBTTL  XI_START,        Start I/O routines
                        010D    ;++
                        010D    ; XI_START - Start a data transfer, set characteristics, enable ATTN AST.
                        010D    ;
                        010D    ; Functional Description:
                        010D    ;
                        010D    ;       This routine has one major function:
                        010D    ;
                        010D    ;       1) Start an I/O transfer. The CTRL bits in the port CSR are set.  If
                        010D    ;          the transfer count is zero, the STATUS bits in the PORT CSR
                        010D    ;          are read and the request completed.
                        010D    ;
                        010D    ; Inputs:
                        010D    ;
                        010D    ;       R3 = Address of the I/O request packet
                        010D    ;       R5 = Address of the UCB
                        010D    ;
                        010D    ; Outputs:
                        010D    ;
                        010D    ;       R0 = final status and number of bytes transferred
                        010D    ;       R1 = value of CSR STATUS bits
                        010D    ;
                        010D    ;--
                        010D
                        010D    XI_START:
                        010D
                        010D    ; Retrieve the address of the device CSR
                        010D
                        010D            ASSUME  IDB$L_CSR EQ 0
      54    24 A5   D0  010D            MOVL    UCB$L_CRB(R5),R4        ; Address of CRB
      54    2C B4   D0  0111            MOVL    @CRB$L_INTD+VEC$L_IDB(R4),R4
                        0115                                            ; Address of CSR
                        0115
                        0115    ; Fetch the I/O function code
                        0115
      51    20 A3   3C  0115            MOVZWL  IRP$W_FUNC(R3),R1       ; Get entire function code
    009A C5    51   B0  0119            MOVW    R1,UCB$W_FUNC(R5)       ; Save FUNC in UCB
 52   51    06 00   EF  011E            EXTZV   #IO$V_FCODE, -
                        0123                    #IO$S_FCODE,R1,R2       ; Extract function field
                        0123
                        0123    ; If subfunction modifier for device reset is set, do one here
                        0123
   03 51       0B   E1  0123            BBC     S^#IO$V_RESET,R1,40$    ; Branch if not device reset
         028B       30  0127            BSBW    XI_DEV_RESET            ; Reset port
                        012A
                        012A    ; This must be a data transfer function - i.e. READ OR WRITE
                        012A    ; Check to see if this is a zero length transfer.
                        012A    ; If so, only set CSR CTRL bits and return STATUS from CSR
                        012A
         7E A5      B5  012A    40$:    TSTW    UCB$W_BCNT(R5)          ; Is transfer count zero?
            50      12  012D            BNEQ    100$                    ; No, continue with data transfer
   3C 51       09   E1  012F            BBC     S^#IO$V_SETFNCT,R1,60$  ; Set CSR CTRL specified?
                        0133            DSBINT                          ; Disable Interrupts
                        0139            SETCTRL                         ; Set CTRL bits in CSR
      51    64      3C  0167            MOVZWL  XI_CSR(R4),R1           ; Save CSR
                        016A            ENBINT                          ; Enable Interrupts
            02      11  016D            BRB     70$                     ; Skip clearing of R1
```

```
                                 016F
                   51     D4     016F   60$:     CLRL     R1                        ; Clear R1
              0060 8F     A8     0171   70$:     BISW     #XI_CSR$M_IEAB,-
                   64            0175                     XI_CSR(R4)                ; Enable device interrupts (A & B)
              50   01     3C     0176            MOVZWL   #SS$_NORMAL,R0            ; Set success
                                 0179            REQCOM                             ; Request done
                                 017F
                                 017F   ;
                                 017F   ; Do the read or the write
                                 017F   ;
                                 017F
                                 017F   100$:
          50      7E A5   3C     017F            MOVZWL   UCB$W_BCNT(R5),R0         ; Get byte count
 00A4 C5  50      FF 8F   78     0183            ASHL     #-1,R0,UCB$L_XI_DPR(R5)   ; Make byte count into word count
                                 018A
                                 018A            .SBTTL  - word mode tranfer
                                 018A   ;++
                                 018A   ; WORD MODE -- Process word mode (interrupt per word) transfer
                                 018A   ;
                                 018A   ; FUNCTIONAL DESCRIPTION:
                                 018A   ;
                                 018A   ;        Data is transferred one word at a time with an interrupt for each word.
                                 018A   ;        The request is handled separately for a write (from memory to port
                                 018A   ;        and a read (from port to memory).
                                 018A   ;        For a write, data is fetched from memory, loaded into the ODR of the
                                 018A   ;        port and the system waits for an interrupt.  For a read, the system
                                 018A   ;        waits for a port interrupt and the INBUF is transferred into memory.
                                 018A   ;        If the unsolicited interrupt flag is set, the first word is transferred
                                 018A   ;        directly into memory withou waiting for an interrupt.
                                 018A   ;--
                                 018A
                                 018A   WORD_MODE:
                                 018A
                                 018A   ; Dispatch to separate loops on READ or WRITE
                                 018A
                                 018A   10$:
              52   0C     91     018A            CMPB     #IO$_READPBLK,R2          ; Check for read function
                   7D     13     018D            BEQL     WORD_MODE_READ
```

```
                              018F  ;++
                              018F  ; WORD MODE WRITE -- Write (output) in word mode
                              018F  ;
                              018F  ; FUNCTIONAL DESCRIPTION:
                              018F  ;
                              018F  ;       Transfer the requested number of words from user memory to
                              018F  ;       the port OUTBUF one word at a time, wait for interrupt for each
                              018F  ;       word.
                              018F  ;--
                              018F
                              018F  WORD_MODE_WRITE:
                              018F  10$:
             0110      30     018F          BSBW    MOVFRUSER                       ; Get two bytes from user buffer
                              0192          DSBINT                                  ; Lock out interrupts
                              0198                                                  ; Flag interrupt expected
          02 A4   51   B0     0198          MOVW    R1,XI_OUTBUF(R4)                ; Move data to port
      64   0060 8F     A8     019C          BISW    #XI_CSR$M_IEAB, -
                              01A1                  XI_CSR(R4)                      ; Set Interrupt Enable (A & B)
                              01A1          SETCTRL                                 ; Clear and set CTRL bits
                              01CF
                              01CF  ; Wait for interrupt, powerfail, or device time-out
                              01CF
                              01CF          WFIKPCH XI_TIME_OUTW,IRP$L_MEDIA(R3)
                              01DA
                              01DA  ; Decrement transfer count, and loop until done
                              01DA
                              01DA          IOFORK                                  ; Fork to lower IPL
             00A4 C5   B7     01E0          DECW    UCB$L_XI_DPR(R5)                ; All words transferred?
                  A9   12     01E4          BNEQ    10$                             ; No, loop until finished.
                              01E6
                              01E6  ; Transfer is done, clear interrupt expected flag and FORK
                              01E6  ; All words read or written in WORD MODE.  Finish I/O.
                              01E6
                              01E6  RETURN_STATUS:
                              01E6
             50   01   3C     01E6          MOVZWL  #SS$_NORMAL,R0                  ; Complete success status
   51   00A4 C5   02   A5     01E9          MULW3   #2,UCB$L_XI_DPR(R5),R1          ; Calculate actual bytes xfered
      51   7E A5  51   A3     01EF          SUBW3   R1,UCB$W_BCNT(R5),R1            ; From requested number of bytes
  50   10   10    51   F0     01F4          INSV    R1,#16,#16,R0                   ; And place in high word of R0
      51   00AA C5     3C     01F9          MOVZWL  UCB$W_XI_CSR(R5),R1             ; Return CSR status
                       AA     01FE          BICW    #<XI_CSR$M_CTRL0! -
                              01FF                    XI_CSR$M_CTRL1>,-
             64   03          01FF                  XI_CSR(R4)                      ; Clear CTRL bits
             0060 8F   A8     0201          BISW    #XI_CSR$M_IEAB,-
                  64          0205                  XI_CSR(R4)                      ; Enable device interrupts (A & B)
                              0206          REQCOM                                  ; Finish request in exec
```

```
                        020C   ;++
                        020C   ; WORD MODE READ -- Read (input) in word mode
                        020C   ;
                        020C   ; FUNCTIONAL DESCRIPTION:
                        020C   ;
                        020C   ;       Transfer the requested number of words from the port INBUF into
                        020C   ;       user memory one word at a time, wait for interrupt for each word.
                        020C   ;       If the unexpected (unsolicited) interrupt bit is set, transfer the
                        020C   ;       first (last received) word to memory without waiting for an
                        020C   ;       interrupt.
                        020C   ;--
                        020C
                        020C   WORD_MODE_READ:
                        020C           SETIPL  UCB$B_DIPL(R5)          ; Lock out interrupts
                        0210
                        0210   ; If an unexpected (unsolicited) interrupt has occured, assume it
                        0210   ; is for this READ request and return value to user buffer without
                        0210   ; waiting for an interrupt.
                        0210
  4A 68 A5    01     E4 0210           BBSC    #UCB$V_UNEXPT, -
                        0215                   UCB$W_DEVSTS(R5),20$    ; Branch if unexpected interrupt
                        0215           DSBINT
   64    0060 8F     A8 021B   10$:    BISW    #XI_CSR$M_IEAB, -
                        0220                   XI_CSR(R4)              ; Set Interrupt Enable (A & B)
                        0220           SETCTRL                         ; Clear and set CTRL bits
                        024E
                        024E   ; Wait for interrupt, powerfail, or device time-out
                        024E
                        024E           WFIKPCH XI_TIME_OUTW,IRP$L_MEDIA(R3)
                        0259
                        0259   ; Decrement transfer count, and loop until done
                        0259
                        0259           IOFORK                          ; Fork to lower IPL
                        025F   20$:
             0051    30 025F           BSBW    MOVTOUSER               ; Store two bytes into user buffer
                        0262
                        0262   ; Send interrupt back to sender.  Acknowledge we got last word.
                        0262
                        0262           DSBINT
          00A4 C5    B7 0268           DECW    UCB$L_XI_DPR(R5)        ; Decrement transfer count
               AD    12 026C           BNEQ    10$                     ; Loop until all words transferred
                        026E           SETCTRL
                        029C           ENBINT
             FF44    31 029F           BRW     RETURN_STATUS           ; Finish request in common code
```

```
                               02A2      ;
                               02A2      ; MOVFRUSER - Routine to fetch two bytes from user buffer.
                               02A2      ;
                               02A2      ; INPUTS:
                               02A2      ;
                               02A2      ;       R5 = UCB address
                               02A2      ;
                               02A2      ; OUTPUTS:
                               02A2      ;
                               02A2      ;       R1 = Two bytes of data from users buffer
                               02A2      ;       Buffer descriptor in UCB is updated.
                               02A2      ;
                               02A2             .ENABL  LSB
                               02A2     MOVFRUSER:
          51     7E      DE    02A2             MOVAL   -(SP),R1                ; Address of temporary stack loc
          52     02      9A    02A5             MOVZBL  #2,R2                   ; Fetch two bytes
    00000000'GF          16    02A8             JSB     G^IOC$MOVFRUSER         ; Call exec routine to do the deed
          51     8E      D0    02AE             MOVL    (SP)+,R1                ; Retrieve the bytes
                 0E      11    02B1             BRB     20$                     ; Update UCB buffer pointers
                               02B3
                               02B3      ;
                               02B3      ; MOVTOUSER - Routine to store two bytes into users buffer.
                               02B3      ;
                               02B3      ; INPUTS:
                               02B3      ;
                               02B3      ;       R5 = UCB address
                               02B3      ;       UCB$W_XI_INBUF(R5) = Location where two bytes are saved
                               02B3      ;
                               02B3      ; OUTPUTS:
                               02B3      ;
                               02B3      ;       Two bytes are stored in user buffer and buffer descriptor in
                               02B3      ;       UCB is updated.
                               02B3      ;
                               02B3
                               02B3     MOVTOUSER:
     51      00A8 C5     9E    02B3             MOVAB   UCB$W_XI_INBUF(R5),R1   ; Address of internal buffer
          52     02      9A    02B8             MOVZBL  #2,R2
    00000000'GF          16    02BB             JSB     G^IOC$MOVTOUSER         ; Call exec
                               02C1     20$:                                    ; Update buffer pointers in UCB
       7C A5     02      A0    02C1             ADDW    #2,UCB$W_BOFF(R5)       ; Add two to buffer descriptor
  7C A5    FE00 8F       AA    02C5             BICW    #^C<^X01FF>,UCB$W_BOFF(R5) ; Modulo the page size
                 04      12    02CB             BNEQ    30$                     ; If NEQ, no page boundary crossed
       78 A5     04      C0    02CD             ADDL    #4,UCB$L_SVAPTE(R5)     ; Point to next page
                               02D1     30$:
                         05    02D1             RSB
                               02D2             .DSABL  LSB
```

```
                                  02D2          .SBTTL  XI_TIME_OUTW,  Device time-out routine
                                  02D2   ;++
                                  02D2   ; Device TIME-OUT
                                  02D2   ;
                                  02D2   ; Clear port CSR
                                  02D2   ; Return error status
                                  02D2   ;
                                  02D2   ; Power failure will appear as a device time-out
                                  02D2   ;--
                                  02D2
                                  02D2   XI_TIME_OUTW:                                       ; Time-out for WORD mode transfer
                                  02D2
                  00E0     30     02D2          BSBW    XI_DEV_RESET                         ; Reset controller
         50   022C 8F      3C     02D5          MOVZWL  #SS$_TIMEOUT,R0                      ; Error status
                    51     D4     02DA          CLRL    R1
               68 A5       B4     02DC          CLRW    UCB$W_DEVSTS(R5)                     ; Clear ATTN AST flags
                           AA     02DF          BICW    #<UCB$M_TIM      ! -
                                  02E0                    UCB$M_INT      ! -
                                  02E0                    UCB$M_TIMOUT   ! -
                                  02E0                    UCB$M_CANCEL   ! -
                                  02E0                    UCB$M_POWER>,-
      64 A5       0068 8F         02E0                  UCB$W_STS(R5)                        ; Clear unit status flags
                                  02E5          REQCOM                                       ; Complete I/O in exec
```

```
                            02EB            .SBTTL  XI_INTERRUPT,    Interrupt service routine for PORT
                            02EB      ;++
                            02EB      ; XI_INTERRUPT, Handles interrupts generated by port
                            02EB      ;
                            02EB      ; Functional description:
                            02EB      ;
                            02EB      ;       This routine is entered whenever an interrupt is generated
                            02EB      ;       by the port.  It checks that an interrupt was expected.
                            02EB      ;       If not, it sets the unexpected (unsolicited) interrupt flag.
                            02EB      ;       All device registers are read and stored into the UCB.
                            02EB      ;       If an interrupt was expected, it calls the driver back at its Wait
                            02EB      ;       For Interrupt point.
                            02EB      ;       Deliver ATTN AST's if unexpected interrupt.
                            02EB      ;
                            02EB      ; Inputs:
                            02EB      ;
                            02EB      ;       00(SP) = Pointer to address of the device IDB
                            02EB      ;       04(SP) = saved R0
                            02EB      ;       08(SP) = saved R1
                            02EB      ;       12(SP) = saved R2
                            02EB      ;       16(SP) = saved R3
                            02EB      ;       20(SP) = saved R4
                            02EB      ;       24(SP) = saved R5
                            02EB      ;       28(SP) = saved PSL
                            02EB      ;       32(SP) = saved PC
                            02EB      ;
                            02EB      ; Outputs:
                            02EB      ;
                            02EB      ;       The driver is called at its Wait For Interrupt point if an
                            02EB      ;       interrupt was expected.
                            02EB      ;       The current value of the port CSR's are stored in the UCB.
                            02EB      ;
                            02EB      ;--
                            02EB      XI_INTERRUPT:                                   ; Interrupt service for PORT
                            02EB
           54     9E    D0  02EB            MOVL    @(SP)+,R4                      ; Address of IDB and pop SP
           54     64    7D  02EE            MOVQ    (R4),R4                        ; CSR and UCB address from IDB
                            02F1      ;
                            02F1      ; Read INBUF and CSR
                            02F1      ;
   00A8 C5     04 A4    B0  02F1            MOVW    XI_INBUF(R4), -
                            02F7                    UCB$W_XI_INBUF(R5)             ; Read input data
                  64    B0  02F7            MOVW    XI_CSR(R4),-
         00AA C5            02F9                    UCB$W_XI_CSR(R5)               ; Read CSR
                            02FC
                            02FC      ; Check to see if device transfer request active or not
                            02FC      ; If so, call driver back at Wait for Interrupt point and
                            02FC      ; Clear unexpected interrupt flag.
                            02FC
     0D 64 A5     01    E5  02FC            BBCC    #UCB$V_INT, -
                            0301                    UCB$W_STS(R5),10$              ; If clear, no interrupt expected
                            0301
                            0301      ; Interrupt expected, clear unexpected interrupt flag and call driver
                            0301      ; back.
                            0301
        68 A5     02    AA  0301            BICW    #UCB$M_UNEXPT, -
                            0305                    UCB$W_DEVSTS(R5)               ; Clear unexpected interrupt flag
```

```
             53    10 A5   D0  0305   1012          MOVL    UCB$L_FR3(R5),R3          ; Restore drivers R3
                   0C B5   16  0309   1013          JSB     @UCB$L_FPC(R5)            ; Call driver back after WFIKPCH
                      0C   11  030C   1014          BRB     20$                       ; Exit
                               030E   1015
                               030E   1016 ; Deliver ATTN AST's if no interrupt expected and set unexpected
                               030E   1017 ; interrupt flag.
                               030E   1018
                               030E   1019 10$:
             68 A5    02   A8  030E   1020          BISW    #UCB$M_UNEXPT, -
                               0312   1021                  UCB$W_DEVSTS(R5)          ; Set unexpected interrupt flag
                    0051   30  0312   1022          BSBW    XI_DEL_ATTNAST            ; Deliver ATTN AST's
                 0060 8F   A8  0315   1023          BISW    #XI_CSR$M_IEAB,-
                      64       0319   1024                  XI_CSR(R4)                ; Enable device interrupts (A & B)
                               031A   1025
                               031A   1026 ; Restore registers and return from interrupt
                               031A   1027
                               031A   1028 20$:
                      3F   BA  031A   1029          POPR    #^M<R0,R1,R2,R3,R4,R5>    ; Restore registers
                           02  031C   1030          REI                               ; Return from interrupt
```

```
                         031D         .SBTTL  XI_CANCEL,       Cancel I/O routine
                         031D   ;++
                         031D   ; XI_CANCEL, Cancels an I/O operation in progress
                         031D   ;
                         031D   ; Functional description:
                         031D   ;
                         031D   ;       Flushes Attention AST queue for the user.
                         031D   ;       If transfer in progress, do a device reset to port
                         031D   ;       and finish the request.
                         031D   ;       Clear interrupt expected flag.
                         031D   ;
                         031D   ; Inputs:
                         031D   ;
                         031D   ;       R2 = negated value of channel index
                         031D   ;       R3 = address of current IRP
                         031D   ;       R4 = address of the PCB requesting the cancel
                         031D   ;       R5 = address of the device's UCB
                         031D   ;
                         031D   ; Outputs:
                         031D   ;
                         031D   ;--
                         031D
                         031D   XI_CANCEL:                                        ; Cancel I/O
                         031D
    1A 68 A5   00   E5   031D           BBCC    #UCB$V_ATTNAST, -
                         0322                   UCB$W_DEVSTS(R5),20$    ; ATTN AST enabled?
                         0322
                         0322   ; Finish all ATTN AST's for this process.
                         0322
       00C4 8F      BB   0322           PUSHR   #^M<R2,R6,R7>
         56 52      D0   0326           MOVL    R2,R6                   ; Set up channel number
    57  00A0 C5     9E   0329           MOVAB   UCB$L_XI_ATTN(R5),R7    ; Address of listhead
     00000000'GF    16   032E           JSB     G^COM$FLUSHATTNS        ; Flush ATTN AST's for process
       00C4 8F      BA   0334           POPR    #^M<R2,R6,R7>
      68 A5   02    AA   0338           BICW    #UCB$M_UNEXPT, -
                         033C                   UCB$W_DEVSTS(R5)        ; Clear unexpected interrupt flag
                         033C
                         033C   ; Check to see if a data transfer request is in progress
                         033C   ; for this process on this channel
                         033C
                         033C   20$:
                         033C           SETIPL  UCB$B_DIPL(R5)          ; Lock out device interrupts
     00000000'GF    16   0340           JSB     G^IOC$CANCELIO          ; Check if transfer going
   16 64 A5   03    E1   0346           BBC     #UCB$V_CANCEL, -
                         034B                   UCB$W_STS(R5),30$       ; Branch if not for this guy
                         034B
     50   0830 8F   3C   034B           MOVZWL  #SS$_CANCEL,R0          ; Status is request canceled
              51    D4   0350           CLRL    R1
           68 A5    B4   0352           CLRW    UCB$W_DEVSTS(R5)        ; Clear unexpected interrupt flag
                    AA   0355           BICW    #<UCB$M_TIM     ! -
                         0356                     UCB$M_BSY     ! -
                         0356                     UCB$M_CANCEL  ! -
                         0356                     UCB$M_INT     ! -
                         0356                     UCB$M_TIMOUT>,-
   64 A5    014B 8F      0356                   UCB$W_STS(R5)           ; Clear unit status flags
                         035B           REQCOM                          ; Jump to exec to finish I/O
                         0361
```

```
                0361  1089 30$:
                0361  1090          SETIPL  UCB$B_FIPL(R5)          ; Lower to FORK IPL
            05  0365  1091          RSB                             ; Return
```

```
                                0366          .SBTTL  XI_DEL_ATTNAST,  Deliver ATTN AST's
                                0366    ;++
                                0366    ; XI_DEL_ATTNAST, Deliver all outstanding ATTN AST's
                                0366    ;
                                0366    ; Functional description:
                                0366    ;
                                0366    ;       This routine is used by the port driver to deliver all of the
                                0366    ;       outstanding attention AST's.  It is copied from COM$DELATTNAST in
                                0366    ;       the exec.  In addition, it places the saved value of the port CSR
                                0366    ;       and Input Data Buffer Register in the AST paramater.
                                0366    ;
                                0366    ; Inputs:
                                0366    ;
                                0366    ;       R5 = UCB of unit
                                0366    ;
                                0366    ; Outputs:
                                0366    ;
                                0366    ;       R0,R1,R2 Destroyed
                                0366    ;       R3,R4,R5 Preserved
                                0366    ;--
                                0366    XI_DEL_ATTNAST:
      49 68 A5    00     E5     0366            BBCC    #UCB$V_ATTNAST, -
                                036B                    UCB$W_DEVSTS(R5),30$     ; Any ATTN AST's expected?
               38     BB        036B            PUSHR   #^M<R3,R4,R5>            ; Save R3,R4,R5
         51    08 AE  D0        036D    10$:    MOVL    8(SP),R1                 ; Get address of UCB
      52    00A0 C1   9E        0371            MOVAB   UCB$L_XI_ATTN(R1),R2     ; Address of ATTN AST listhead
            55    62  D0        0376            MOVL    (R2),R5                  ; Address of next entry on list
               37     13        0379            BEQL    20$                      ; No next entry, end of loop
       68 A1   02     AA        037B            BICW    #UCB$M_UNEXPT, -
                                037F                    UCB$W_DEVSTS(R1)         ; Clear unexpected interrupt flag
            62    65  D0        037F            MOVL    (R5),(R2)                ; Close list
   1E A5  00A8 C1     B0        0382            MOVW    UCB$W_XI_INBUF(R1), -
                                0388                    ACB$L_KAST+6(R5)         ; Store INBUF in AST paramater
   1C A5  00AA C1     B0        0388            MOVW    UCB$W_XI_CSR(R1), -
                                038E                    ACB$L_KAST+4(R5)         ; Store CSR in AST paramater
          DC AF       9F        038E            PUSHAB  B^10$                    ; Set return address for FORK
                                0391                                             ;   so that it loops through all AST's
                                0391            FORK                             ; FORK for this AST
                                0397
                                0397    ; AST fork procedure
                                0397
   10 A5   18 A5      7D        0397            MOVQ    ACB$L_KAST(R5),ACB$L_AST(R5)
                                039C                                             ; Re-arrange entries
   0B A5   20 A5      90        039C            MOVB    ACB$L_KAST+8(R5),ACB$B_RMOD(R5)
   0C A5   24 A5      D0        03A1            MOVL    ACB$L_KAST+12(R5),ACB$L_PID(R5)
           18 A5      D4        03A6            CLRL    ACB$L_KAST(R5)
         52    01     9A        03A9            MOVZBL  #PRI$_IOCOM,R2           ; Set up priority increment
     00000000'GF      17        03AC            JMP     G^SCH$QAST               ; Queue the AST
                                03B2
               38     BA        03B2    20$:    POPR    #^M<R3,R4,R5>            ; Restore registers
                      05        03B4    30$:    RSB                              ; Return
```

```
                 03B5                .SBTTL  XI_DEV_RESET,    Device reset routine
                 03B5        ;++
                 03B5        ; XI_DEV_RESET - Device reset routine
                 03B5        ;
                 03B5        ; This routine raises IPL to device IPL, performs a device reset to
                 03B5        ; the required controler, and re-enables device interrupts.
                 03B5        ;
                 03B5        ; Inputs:
                 03B5        ;
                 03B5        ;       R4 - Address of Control and Status Register
                 03B5        ;       R5 - Address of UCB
                 03B5        ;
                 03B5        ; Outputs:
                 03B5        ;
                 03B5        ;       Controller is reset, controller interrupts are enabled
                 03B5        ;
                 03B5        ;--
                 03B5
                 03B5        XI_DEV_RESET:
                 03B5
                 03B5                DSBINT                          ; Raise IPL to lock all interrupts
                 03BB
4000 8F     A8   03BB                BISW    #XI_CSR$M_RESET,-
     64          03BF                        XI_[CSR(R4]             ; Reset device
                 03C0
                 03C0                TIMEWAIT -                      ; Timewait to allow reset
                 03C0                        TIME = #500,-
                 03C0                        BITVAL = #XI_CSR$M_RESET,-
                 03C0                        SOURCE = XI_[CSR(R4],-
                 03C0                        CONTEXT = W,-
                 03C0                        SENSE = .FALSE.
                 03E9
0060 8F     A8   03E9                BISW    #XI_CSR$M_IEAB,-
     64          03ED                        XI_[CSR(R4]             ; Enable device interrupts (A & B)
                 03EE
                 03EE                ENBINT                          ; Restore IPL
            05   03F1                RSB
                 03F2
                 03F2        XI_END:                                 ; End of driver label
                 03F2                .END
```

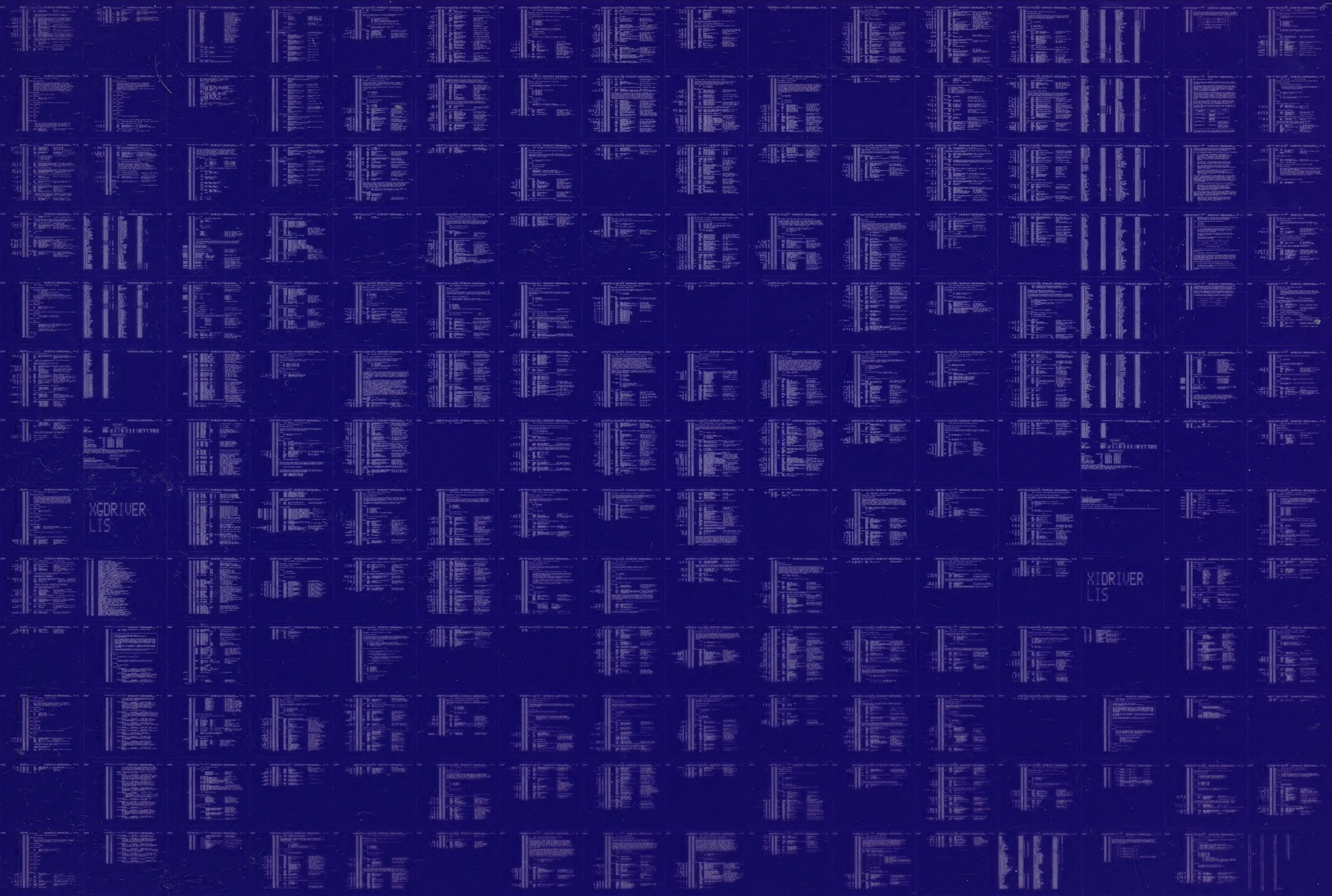
```
                                                             C  1
XIDRIVER                          - VAX/VMS DMF32 PARALLEL PORT DRIVER     16-SEP-1984 00:16:11  VAX/VMS Macro V04-00        Page  30
Symbol table                                                                6-SEP-1984 16:33:12  [DRIVER.SRC]XIDRIVER.MAR;2      (18)

$$$                          = 00000020 R    02     IO$_SETMODE              = 00000023
$$OP                         = 00000002             IO$_VIRTUAL              = 0000003F
ACB$B_RMOD                   = 0000000B             IO$_WRITELBLK            = 00000020
ACB$L_AST                    = 00000010             IO$_WRITEPBLK            = 0000000B
ACB$L_KAST                   = 00000018             IO$_WRITEVBLK            = 00000030
ACB$L_PID                    = 0000000C             IOC$CANCELIO               ********   X   03
AT$_UBA                      = 00000001             IOC$MNTVER                 ********   X   03
COM$FLUSHATTNS                 ********   X  03     IOC$MOVFRUSER              ********   X   03
COM$SETATTNAST                 ********   X  03     IOC$MOVTOUSER              ********   X   03
CRB$L_INTD                   = 00000024             IOC$REQCOM                 ********   X   03
CRB$L_INTD2                  = 00000048             IOC$RETURN                 ********   X   03
DC$_REALTIME                 = 00000060             IOC$WFIKPCH                ********   X   03
DDB$L_DDT                    = 0000000C             IRP$L_MEDIA              = 00000038
DEV$M_AVL                      ********   X  02     IRP$L_SEGVBN             = 00000048
DEV$M_IDV                      ********   X  02     IRP$W_FUNC               = 00000020
DEV$M_ODV                      ********   X  02     MASKH                    = 00000080
DEV$M_RTM                      ********   X  02     MASKL                    = 08000000
DPT$C_LENGTH                 = 00000038             MOVFRUSER                  000002A2 R     03
DPT$C_VERSION                = 00000004             MOVTOUSER                  000002B3 R     03
DPT$INITAB                     00000038 R    02     P1                       = 00000000
DPT$M_SVP                    = 00000002             P2                       = 00000004
DPT$REINITAB                   00000054 R    02     P3                       = 00000008
DPT$TAB                        00000000 R    02     P4                       = 0000000C
DT$_XI_DR11C                 = 0000000D             P5                       = 00000010
DYN$C_CRB                    = 00000005             P6                       = 00000014
DYN$C_DDB                    = 00000006             PR$_IPL                  = 00000012
DYN$C_DPT                    = 0000001E             PRI$_IOCOM               = 00000001
DYN$C_UCB                    = 00000010             RETURN_STATUS              000001E6 R     03
EXE$ABORTIO                    ********   X  03     SCH$QAST                   ********   X   03
EXE$FINISHIO                   ********   X  03     SIZ...                   = 00000001
EXE$FORK                       ********   X  03     SS$_BADPARAM             = 00000014
EXE$GL_TENUSEC                 ********   X  03     SS$_CANCEL               = 00000830
EXE$GL_UBDELAY                 ********   X  03     SS$_NORMAL               = 00000001
EXE$IOFORK                     ********   X  03     SS$_TIMEOUT              = 0000022C
EXE$READ                       ********   X  03     UCB$B_DEVCLASS           = 00000040
EXE$SENSEMODE                  ********   X  03     UCB$B_DEVTYPE            = 00000041
EXE$SETCHAR                    ********   X  03     UCB$B_DIPL               = 0000005E
EXE$WRITE                      ********   X  03     UCB$B_FIPL               = 0000000B
FUNCTAB_LEN                  = 0000004C             UCB$K_SIZE               = 000000AC
IDB$B_COMBO_CSR_OFFSET       = 0000000F             UCB$L_CRB                = 00000024
IDB$B_COMBO_VECTOR_OFFSET    = 00000010             UCB$L_DEVCHAR            = 00000038
IDB$B_VECTOR                 = 0000000B             UCB$L_DPC                = 0000009C
IDB$L_CSR                    = 00000000             UCB$L_FPC                = 0000000C
IDB$L_OWNER                  = 00000004             UCB$L_FR3                = 00000010
IDB$L_UCBLST                 = 00000018             UCB$L_SVAPTE             = 00000078
IO$S_FCODE                   = 00000006             UCB$L_XI_ATTN              000000A0
IO$V_ATTNAST                 = 00000008             UCB$L_XI_DPR               000000A4
IO$V_FCODE                   = 00000000             UCB$M_ATTNAST            = 00000001
IO$V_RESET                   = 00000008             UCB$M_BSY                = 00000100
IO$V_SETFNCT                 = 00000009             UCB$M_CANCEL             = 00000008
IO$V_TIMED                   = 00000007             UCB$M_INT                = 00000002
IO$_READLBLK                 = 00000021             UCB$M_ONLINE             = 00000010
IO$_READPBLK                 = 0000000C             UCB$M_POWER              = 00000020
IO$_READVBLK                 = 00000031             UCB$M_TIM                = 00000001
IO$_SENSECHAR                = 0000001B             UCB$M_TIMOUT             = 00000040
IO$_SENSEMODE                = 00000027             UCB$M_UNEXPT             = 00000002
IO$_SETCHAR                  = 0000001A             UCB$V_ATTNAST            = 00000000
```

```
UCB$V_CANCEL                    = 00000003
UCB$V_INT                       = 00000001
UCB$V_POWER                     = 00000005
UCB$V_UNEXPT                    = 00000001
UCB$W_BCNT                      = 0000007E
UCB$W_BOFF                      = 0000007C
UCB$W_DEVBUFSIZ                 = 00000042
UCB$W_DEVSTS                    = 00000068
UCB$W_FUNC                      = 0000009A
UCB$W_STS                       = 00000064
UCB$W_XI_CSR                      000000AA
UCB$W_XI_INBUF                    000000A8
VEC$B_DATAPATH                  = 00000013
VEC$L_IDB                       = 00000008
VEC$L_INITIAL                   = 0000000C
VEC$M_PATHLOCK                  = 00000080
WORD_MODE                         0000018A R       03
WORD_MODE_READ                    0000020C R       03
WORD_MODE_WRITE                   0000018F R       03
XI$DDT                            00000000 RG      03
XI$K_VEC_OFFSET                 = 00000002
XI_CANCEL                         0000031D R       03
XI_CONTROL_INIT                   00000084 R       03
XI_CSR                            00000000
XI_CSR$M_CTRL0                  = 00000001
XI_CSR$M_CTRL1                  = 00000002
XI_CSR$M_IEAB                   = 00000060
XI_CSR$M_INTENB_A               = 00000020
XI_CSR$M_INTENB_B               = 00000040
XI_CSR$M_RESET                  = 00004000
XI_DEF_BUFSIZ                   = 0000FFFF
XI_DEF_TIMEOUT                  = 0000000A
XI_DEL_ATTNAST                    00000366 R       03
XI_DEV_RESET                      000003B5 R       03
XI_END                            000003F2 R       03
XI_FUNCTABLE                      00000038 R       03
XI_INBUF                          00000004
XI_IND                            00000006
XI_INTERRUPT                      000002EB R       03
XI_MISC                           00000004
XI_OUTBUF                         00000002
XI_READ_WRITE                     000000A9 R       03
XI_SETMODE                        000000CF R       03
XI_START                          0000010D R       03
XI_TIME_OUTW                      000002D2 R       03

                                                   +-----------------+
                                                   ! Psect synopsis  !
                                                   +-----------------+

PSECT name                      Allocation         PSECT No.  Attributes
----------                      ----------         ---------  ----------
. ABS  .                        00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           000000AC  (  172.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$105_PROLOGUE                 00000069  (  105.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$115_DRIVER                   000003F2  ( 1010.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD     WRT NOVEC LONG
```

```
                                        +-------------------------+
                                        ! Performance indicators !
                                        +-------------------------+

Phase                       Page faults    CPU Time        Elapsed Time
-----                       -----------    --------        ------------
Initialization                       30    00:00:00.04     00:00:00.71
Command processing                  106    00:00:00.39     00:00:02.80
Pass 1                              496    00:00:14.23     00:00:51.93
Symbol table sort                     0    00:00:02.05     00:00:06.46
Pass 2                              211    00:00:03.10     00:00:10.70
Symbol table output                  20    00:00:00.12     00:00:00.59
Psect synopsis output                 1    00:00:00.01     00:00:00.10
Cross-reference output                0    00:00:00.00     00:00:00.00
Assembler run totals                866    00:00:19.94     00:01:13.30

The working set limit was 1950 pages.
118674 bytes (232 pages) of virtual memory were used to buffer the intermediate code.
There were 110 pages of symbol table space allocated to hold 1953 non-local and 39 local symbols.
1184 source lines were read in Pass 1, producing 18 object records in Pass 2.
40 pages of virtual memory were used to define 37 macros.

                                        +----------------------------+
                                        ! Macro library statistics !
                                        +----------------------------+

Macro library name                            Macros defined
------------------                            --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                            24
_$255$DUA28:[SYSLIB]STARLET.MLB;2                          9
TOTALS (all libraries)                                    33

2206 GETS were required to define 33 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:XIDRIVER/OBJ=OBJ$:XIDRIVER MSRC$:XIDRIVER/UPDATE=(ENH$:XIDRIVER)+EXECML$/LIB
```

0120 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0121 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

XQDRIVER
LIS

XMDRIVER
LIS